AN OPTIMAL ALGORITHM FOR COMPUTING ANGLE-CONSTRAINED SPANNERS*

Paz Carmi,† *and Michiel Smid*‡

Abstract. Let S be a set of n points in $\mathbb{R}^d$ and let $t > 1$ be a real number. A graph $G = (S, E)$ is called a t-spanner for S, if for any two points p and q in S, the shortest-path distance in G between p and q is at most $t|pq|$, where $|pq|$ denotes the Euclidean distance between p and q. The graph G is called θ-angle-constrained, if any two distinct edges sharing an endpoint make an angle of at least θ. It is shown that, for any θ with $0 < \theta < \pi/3$, a θ-angle-constrained t-spanner can be computed in $O(n \log n)$ time, where t depends only on θ. For values of θ approaching 0, we have $t = 1 + O(\theta)$.

1 Introduction

Let S be a set of n points in $\mathbb{R}^d$, where $d \geq 1$ is a constant, and let $G = (S, E)$ be a graph with vertex set S, in which the length (or weight) of every edge $\{p, q\}$ is equal to the Euclidean distance $|pq|$ between p and q. The length of a path in G is defined to be the sum of the lengths of the edges on the path. For any two points p and q in S, denote by $\delta_G(p, q)$ the minimum length of any path in G between p and q. For a real number $t > 1$, G is a *t-spanner* for S, if $\delta_G(p, q) \leq t|pq|$ for any two points p and q of S. The smallest t for which G is a t-spanner is called the *stretch factor* of G.

The problem of efficiently constructing spanners for a given point set has been well-studied. For any set S of n points in $\mathbb{R}^d$ and any constant $t > 1$, a t-spanner for S with $O(n)$ edges can be computed in $O(n \log n)$ time; see Salowe [20], Vaidya [23], and Callahan and Kosaraju [6]. Chen *et al.* [8] have shown an $\Omega(n \log n)$ lower bound in the algebraic computation-tree model for any spanner construction algorithm. For overviews of the main results for geometric spanners, we refer to Eppstein [13] and Narasimhan and Smid [19].

Geometric spanners have many applications in various domains and, hence, have received much attention in the wireless network community; see, e.g., the book [16] by Li. For a positive real number θ, we say that the graph $G = (S, E)$ is *θ-angle-constrained*, if for any two distinct edges $\{p, q\}$ and $\{p, r\}$ in E, the angle $\angle(pq, pr)$ between them is at least θ. On page 238 in his book, Li mentions that a wireless network being an angle-constrained spanner is a desirable property, because it reduces signal interference and receiving power

*MS was supported by NSERC. A preliminary version of this paper appeared in the Proceedings of the 21st Annual International Symposium on Algorithms and Computation (ISAAC), Part I, Lecture Notes in Computer Science, Vol. 6506, Springer-Verlag, Berlin, 2010, pp. 316–327.

†*Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva, Israel*, carmip@cs.bgu.ac.il

‡*School of Computer Science, Carleton University, Ottawa, Canada*, michiel@scs.carleton.ca

cost when directional antennae are used, and it guarantees short paths between any pair of nodes.

In this paper, we consider the problem of computing angle-constrained spanners for point sets in $\mathbb{R}^d$.

The "path-greedy" algorithm of Althöfer *et al.* [1] is a well-known algorithm for constructing a t-spanner. Soares [22] has shown that this spanner is θ-angle-constrained, where θ depends on t. However, the fastest known algorithm for computing the greedy spanner has a running time of $O(n^2 \log n)$; see Bose *et al.* [5]. In [21], Salowe presents an "angle-greedy" algorithm that constructs an angle-constrained spanner; this algorithm, however, also has a running time of $O(n^2 \log n)$. The "gap-greedy" algorithm of Arya and Smid [3] can be modified so that it constructs an angle-constrained spanner in $O(n \log^d n)$ time.

There are $O(n \log n)$–time algorithms that construct, for any constant $t > 1$, a t-spanner whose maximum degree is bounded by a constant; see Arya *et al.* [2]. Salowe [21] showed that, for a sufficiently large value of t, a t-spanner of maximum degree 4 can be computed in $O(n \log n)$ time. The degree-bound was improved to 3 by Das and Heffernan [11]. In dimension $d = 2$, there are $O(n \log n)$–time algorithms that construct, for a sufficiently large value of t, a plane t-spanner whose maximum degree is bounded by a constant; see Bonichon *et al.* [4] for the currently best degree-bound of six. None of these algorithms, however, produces a graph that is angle-constrained.

In this paper, we prove that angle-constrained spanners can be constructed in $O(n \log n)$ time:

Theorem 1. *Let S be a set of n points in $\mathbb{R}^d$, where $d \geq 1$ is a constant, and let θ and ϵ be two real numbers such that $0 < \theta < \pi/3$ and $0 < \epsilon < (\pi - 3\theta)/(21 + \pi)$. In*

$$O\left(\left(1/\epsilon^{2d}\right)(\log(1/\epsilon))n \log n + \left(1/\epsilon^{3d}\right) n\right)$$

time, a θ-angle-constrained t-spanner for S can be computed, where

$$t = \frac{1 + \sqrt{2(1+\epsilon)(1 - \cos(\theta + 7\epsilon)) + \epsilon^2}}{2\cos(\theta + 7\epsilon) - 1 - \epsilon}.$$

Note that this result depends on two parameters: The real number θ determines a lower bound on the minimum angle between any two edges that share a vertex, the real number ϵ determines the constant factor in the running time, whereas the upper bound t on the stretch factor is a function of both parameters.

If we want a t-spanner with t approaching 1, then we take θ and ϵ to be equal and let it approach 0. Using the approximations $\cos x \approx 1 - x^2/2$ and $\sqrt{1+x} \approx 1 + x/2$, which are valid for small positive real numbers x, the upper bound t on the stretch factor becomes $1 + O(\theta)$. Since the t-spanner is θ-angle-constrained, its maximum degree is $O(1/\theta^{d-1}) = O(1/(t-1)^{d-1})$; see Theorem 5.3.1 in [19]. This gives the following result:

Corollary 1. Let S be a set of n points in $\mathbb{R}^d$, where $d \geq 1$ is a constant, and let $t > 1$ be a real number. In

$$O\left(\left(1/(t-1)^{2d}\right)(\log(1/(t-1)))n \log n + \left(1/(t-1)^{3d}\right) n\right)$$

time, a t-spanner for S can be computed, whose maximum degree is $O(1/(t-1)^{d-1})$.

We remark that the previously best-known upper bound on the maximum degree (for $O(n \log n)$–time algorithms) was $O(1/(t-1)^{2d-1})$; see Section 10.1 in [19]. Let S be the set in $\mathbb{R}^d$ consisting of the origin and $n-1$ points evenly spaced on the unit-sphere. Elkin and Solomon [12] have shown that any t-spanner for S has maximum degree $\Omega(1/(t-1)^{d-1})$. Therefore, the result in Corollary 1 is optimal.

1.1 Organization of the Paper

Our construction will be based on a combination of the spanner based on the well-separated pair decomposition of Callahan and Kosaraju [6, 7] (see also Section 2) and ideas that have been used in analyzing the Θ-graph spanner of Clarkson [9] and Keil and Gutwin [15]. As we will show in Section 3, this combination leads to a simple and sufficient condition for a graph being a spanner. In order to satisfy this condition, our algorithms will use simplicial cones, which are described in Section 4. In Section 5, we will present a simple algorithm that constructs an angle-constrained graph that satisfies the condition in Section 3; thus, it produces an angle-constrained spanner. The running time of this algorithm is, however, $O(n \log^{d-1} n)$. Moreover, it does not work in the algebraic computation-tree model. We decided to include this non-optimal algorithm, because it is a good illustration of the general approach that is used to obtain an angle-constrained spanner. We also believe that it makes it easier to read and understand the final algorithm in Section 6. In that section, we will show that the algorithm of Section 5 can be modified such that it works in the algebraic computation-tree model and its running time is $O(n \log n)$. The main ingredient of this final algorithm is the use of the "dumbbell trees" of Arya *et al.* [2].

2 Well-Separated Pairs

Our algorithms will use the well-separated pair decomposition of Callahan and Kosaraju [7]. In this section, we briefly review this decomposition.

For any point set A in $\mathbb{R}^d$, denote its *bounding box* by $R(A)$; thus, $R(A)$ is the smallest axes-parallel hyperrectangle that contains all points of A.

Let $s > 0$ be a real number, which we call the *separation ratio*. Two point sets A and B in $\mathbb{R}^d$ are *s-well-separated*, if there exist two balls of the same radius, say, ρ, one ball containing $R(A)$ and the other ball containing $R(B)$, such that the distance between the balls is at least $s\rho$. The following lemma follows from this definition:

Lemma 1. Let A and B be two sets of points that are s-well-separated, let a, a', and a'' be points in $R(A)$, and let b, b', and b'' be points in $R(B)$. Then the following inequalities hold:

1. $|aa'| \leq (2/s)|a''b''|$,
2. $|bb'| \leq (2/s)|a''b''|$,

3. $|ab| \leq (1 + 4/s)|a'b'|$.

Let S be a set of n points in $\mathbb{R}^d$ and let $s > 0$ be a real number. A *well-separated pair decomposition (WSPD)* of S, with separation ratio s, is a sequence $\{A_1, B_1\}, \ldots, \{A_m, B_m\}$ of s-well-separated pairs of subsets of S, such that, for any two distinct points p and q in S, there is a unique index i such that $p \in A_i$ and $q \in B_i$ or $p \in B_i$ and $q \in A_i$. We will refer to the number m of pairs as the *size* of the WSPD.

Theorem 2 (Callahan and Kosaraju [7])**.** *Let S be a set of n points in $\mathbb{R}^d$ and let $s > 0$ be a real number. A WSPD, with separation ratio s, of size $O(s^d n)$ can be computed in $O(n \log n + s^d n)$ time.*

The algorithm of Callahan and Kosaraju uses the so-called fair-split tree, which is a binary tree storing the points of S at its leaves. For each pair $\{A_i, B_i\}$ in the WSPD, there are two nodes u and v in this tree such that A_i is the set of all points stored in u's subtree and B_i is the set of all points stored in v's subtree. It follows from their construction that $A_i = S \cap R(A_i)$ and $B_i = S \cap R(B_i)$.

3 A Sufficient Condition for Being a Spanner

In this section, we introduce a general property which implies that a geometric graph is a spanner. The property is based on a combination of the WSDP-spanner of [6] and techniques that have been used in the analysis of the Θ-graph spanner of [9, 15].

We fix real numbers α, λ, ϵ, and s, such that

$$0 < \alpha < \pi/3, \ \lambda \geq 1, \ 0 < \epsilon < 2\cos\alpha - 1, \text{ and } s > 8\lambda. \tag{1}$$

Let S be a set of n points in $\mathbb{R}^d$ and consider a WSPD $\{A_1, B_1\}, \ldots, \{A_m, B_m\}$ of S with separation ratio s. Let $G = (S, E)$ be a graph with vertex set S. For any i with $1 \leq i \leq m$, consider the following three properties P.1, P.2, and P.3, which are illustrated in Figure 1:

P.1: For every point p in A_i, the edge set E contains an edge $\{p, r\}$ such that for every point q in B_i, $\angle(pq, pr) \leq \alpha$ and $|pr| \leq (1 + \epsilon)|pq|$.

P.2: For every point q in B_i, the edge set E contains an edge $\{q, r\}$ such that for every point p in A_i, $\angle(qp, qr) \leq \alpha$ and $|qr| \leq (1 + \epsilon)|pq|$.

P.3: Let ℓ_i be the distance between the centers of $R(A_i)$ and $R(B_i)$. The edge set E contains an edge $\{x, y\}$, such that for every point p in A_i and every point q in B_i, both $|px|$ and $|qy|$ are at most $(2\lambda/s)\ell_i$.

Informally, property P.1 states that every point p in A_i has an edge $\{p, r\}$, such that the line segment pr takes us "in the direction of" B_i and does not take us "too far beyond" B_i. Property P.2 is symmetric to P.1. Finally, property P.3 states that there exists an edge $\{x, y\}$, where x is "close" to A_i and y is "close" to B_i.

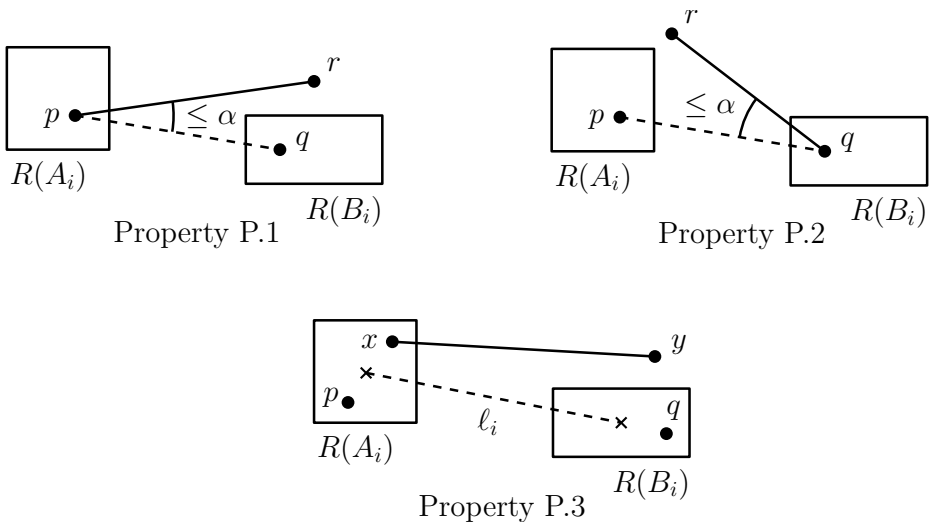

Figure 1: *Illustrating properties P.1, P.2, and P.3.*

In Lemma 4 below, we will prove that the graph G is a spanner, provided that for each pair $\{A_i, B_i\}$, at least one of P.1, P.2, and P.3 holds. The proof of this lemma will use the following two technical results.

Lemma 2. We have

$$1 + \epsilon < \frac{1 + \sqrt{2(1+\epsilon)(1-\cos\alpha) + \epsilon^2}}{2\cos\alpha - 1 - \epsilon}.$$

Proof. Put $a = 1 + \epsilon$. Since $(a-1)^2 > 0$, which is equivalent to $2a - a^2 < 1$, we have

$$2a\cos\alpha - a^2 \leq 2a - a^2 < 1 < 1 + \sqrt{2a(1-\cos\alpha) + (a-1)^2}.$$

Using the fact (see (1)) that $2\cos\alpha - a > 0$, we obtain

$$a < \frac{1 + \sqrt{2a(1-\cos\alpha) + (a-1)^2}}{2\cos\alpha - a}.$$

□

Lemma 3. Let t be a real number such that

$$t \geq \frac{1 + \sqrt{2(1+\epsilon)(1-\cos\alpha) + \epsilon^2}}{2\cos\alpha - 1 - \epsilon}. \tag{2}$$

Let p, q, and r be three distinct points in $\mathbb{R}^d$, such that $\angle(pq, pr) \leq \alpha$ and $|pr| \leq (1+\epsilon)|pq|$. Then

1. $|rq| < |pq|$ and
2. $|pr| + t|rq| \leq t|pq|$.

Proof. Let $\beta = \angle(pq, pr)$. By the Law of Cosines, we have

$$|rq|^2 = |pq|^2 + |pr|^2 - 2|pq||pr|\cos\beta.$$

Thus, $|rq| < |pq|$ if and only if

$$|pr| < 2|pq|\cos\beta. \tag{3}$$

Since $|pr| \leq (1+\epsilon)|pq|$ and $\cos\alpha \leq \cos\beta$, the inequality in (3) holds if $1+\epsilon < 2\cos\alpha$. By our choice of α and ϵ (see (1)), the latter inequality holds. Thus, we have shown that $|rq| < |pq|$.

To prove the second claim, we have

$$|pr| + t|rq| = |pr| + t\sqrt{|pq|^2 + |pr|^2 - 2|pq||pr|\cos\beta}.$$

Thus, $|pr| + t|rq| \leq t|pq|$ if and only if

$$t\sqrt{|pq|^2 + |pr|^2 - 2|pq||pr|\cos\beta} \leq t|pq| - |pr|. \tag{4}$$

By Lemma 2, we have $|pr| \leq (1+\epsilon)|pq| \leq t|pq|$. Thus, $t|pq| - |pr| \geq 0$, implying that (4) is equivalent to

$$t^2\left(|pq|^2 + |pr|^2 - 2|pq||pr|\cos\beta\right) \leq \left(t|pq| - |pr|\right)^2,$$

which, in turn, is equivalent to

$$\left(t^2 - 1\right)|pr| \leq 2t\left(t\cos\beta - 1\right)|pq|.$$

Since $|pr| \leq (1+\epsilon)|pq|$ and $\cos\alpha \leq \cos\beta$, it suffices to show that

$$\left(t^2 - 1\right)(1+\epsilon) \leq 2t\left(t\cos\alpha - 1\right),$$

which can be rewritten as

$$\left(2\cos\alpha - 1 - \epsilon\right)t^2 - 2t + (1+\epsilon) \geq 0. \tag{5}$$

By considering the left-hand side as a quadratic function of t, we see that the largest root of this function is equal to the right-hand side in (2). Thus, by our choice of t, the inequality in (5) holds. □

Lemma 4. Assume that for each i with $1 \leq i \leq m$, at least one of the properties P.1, P.2, and P.3 is satisfied. Then for any real number t with

$$t \geq \max\left(\frac{1 + \sqrt{2(1+\epsilon)(1-\cos\alpha) + \epsilon^2}}{2\cos\alpha - 1 - \epsilon}, \frac{s + 8\lambda}{s - 8\lambda}\right),$$

the graph G is a t-spanner for S.

Proof. We have to show that $\delta_G(p,q) \leq t|pq|$ for all p and q in S. The proof is by induction on the rank of the distance $|pq|$ in the sorted sequence of distances in S. If $p = q$, then the claim obviously holds. Let $p \neq q$ and assume that $\delta_G(a,b) \leq t|ab|$ for all a and b in S with $|ab| < |pq|$. Let i be the index such that (i) $p \in A_i$ and $q \in B_i$ or (ii) $p \in B_i$ and $q \in A_i$. We may assume without loss of generality that (i) holds. There are three possible cases.

Case 1: Property P.1 holds for the pair $\{A_i, B_i\}$.

Consider the edge $\{p,r\}$ in property P.1. By the first claim in Lemma 3, we have $|rq| < |pq|$. Thus, by using the induction hypothesis, our choice for t, and the second claim in Lemma 3, we obtain

$$\delta_G(p,q) \leq |pr| + \delta_G(r,q) \leq |pr| + t|rq| \leq t|pq|.$$

Case 2: Property P.2 holds for the pair $\{A_i, B_i\}$.

This case is symmetric to Case 1.

Case 3: Property P.3 holds for the pair $\{A_i, B_i\}$.

Consider the edge $\{x,y\}$ in property P.3. Using this property and Lemma 1, we have

$$|px| \leq (2\lambda/s)\ell_i \leq (2\lambda/s)(1+4/s)|pq|.$$

Since $s > 4$, we have $1 + 4/s < 2$, and, therefore,

$$|px| < (4\lambda/s)|pq| < |pq|,$$

where the last inequality uses the fact that $s > 4\lambda$. By a symmetric argument, we have

$$|qy| < (4\lambda/s)|pq| < |pq|.$$

Thus, by using the induction hypothesis, we obtain

$$\delta_G(p,q) \leq \delta_G(p,x) + |xy| + \delta_G(y,q) \leq t|px| + |xy| + t|yq|.$$

Since $|xy| \leq |xp| + |pq| + |qy|$, it follows that

$$\begin{aligned} \delta_G(p,q) &\leq (t+1)|px| + |pq| + (t+1)|qy| \\ &\leq (t+1)(4\lambda/s)|pq| + |pq| + (t+1)(4\lambda/s)|pq|. \end{aligned}$$

By our choice of t, the latter quantity is at most $t|pq|$. □

In the rest of this section, we give an informal description of our algorithm. Let θ be a real number with $0 < \theta < \pi/3$. We choose a real number $\epsilon > 0$ such that $\alpha := \theta + O(\epsilon) < \pi/3$. We also choose a real number $\lambda \geq 1$. Let $\mathcal{C}$ be a collection of cones[1] that cover $\mathbb{R}^d$, such that for each cone C in $\mathcal{C}$, the apex of C is at the origin 0 and

$$\max\{\angle(0x, 0y) : x, y \in C \setminus \{0\}\} \leq \epsilon.$$

[1] See Section 4 for a formal definition.

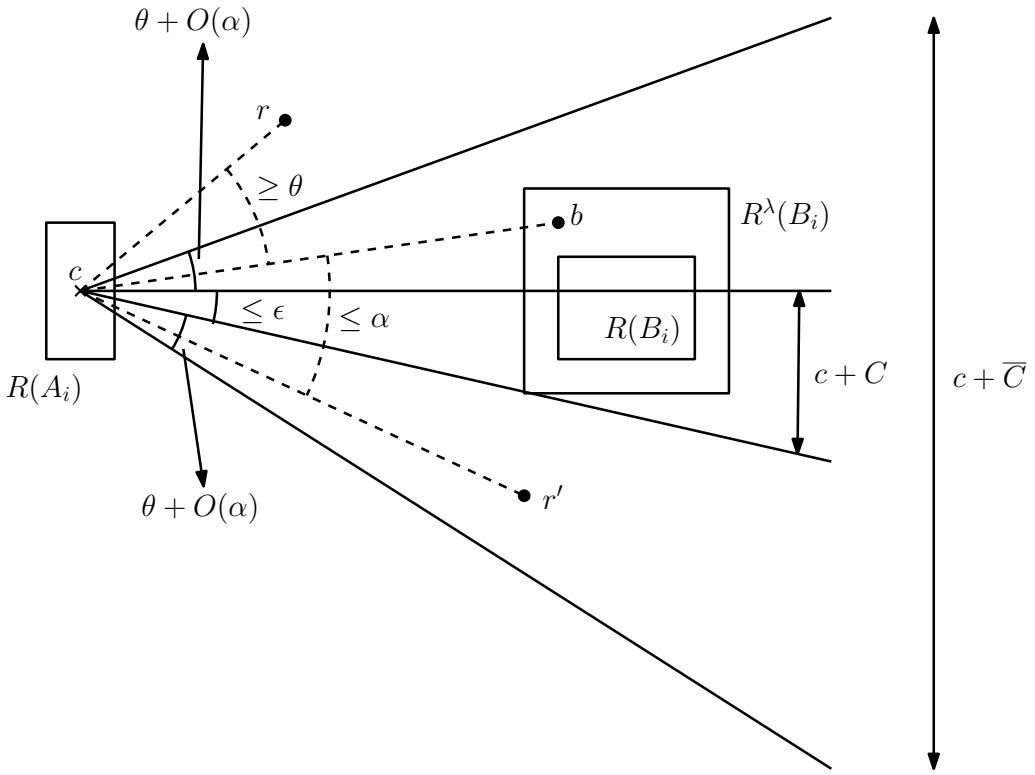

Figure 2: *Illustrating the main properties of the cones* $\overline{C}$.

For any point c in $\mathbb{R}^d$ and any cone C in $\mathcal{C}$, let $c + C$ denote the cone with apex c obtained by translating C, i.e.,

$$c + C = \{c + x : x \in C\}.$$

For each cone C in $\mathcal{C}$, define $\overline{C}$ to be the union of all cones C' in $\mathcal{C}$ that make an angle of at most $\theta + O(\epsilon)$ with C.

Consider a WSPD $\{A_1, B_1\}, \ldots, \{A_m, B_m\}$ of the point set S with separation ratio s. For each i, let ℓ_i be the distance between the centers of $R(A_i)$ and $R(B_i)$. Let $R^\lambda(A_i)$ and $R^\lambda(B_i)$ be boxes of diameter $O(\lambda \ell_i / s)$ that contain $R(A_i)$ and $R(B_i)$, respectively.

We choose the separation ratio s large enough such that for any point x in $R^\lambda(A_i)$ and any two points y and y' in $R^\lambda(B_i)$, $\angle(xy, xy') \leq \epsilon$.

Consider a pair $\{A_i, B_i\}$ in the WSPD. Let c be the center of $R(A_i)$ and let C be a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps $c + C$. Consider the corresponding cone $\overline{C}$. The parameters will be chosen such that the following two properties hold (refer to Figure 2):

1. For any point $r \notin c + \overline{C}$ and any point $b \in R^\lambda(B_i)$, the angle between cr and cb is at least θ.
2. For any point $r' \in c + \overline{C}$ and any point $b \in R^\lambda(B_i)$, the angle between cr' and cb is at most α.

Assume that the pairs in the WSPD have been sorted such that $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$. The algorithm will start with an empty edge set E. Then it processes each pair in the WSPD. Consider the current pair $\{A_i, B_i\}$.

Let c be the center of $R(A_i)$ and let C be a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps $c+C$. Let c' be the center of $R(B_i)$ and let C' be a cone in $\mathcal{C}$ such that $R(A_i)$ overlaps $c'+C'$. There are three possible cases:

1. If every point p in A_i is incident on some edge $\{p, r\}$ with $r \in p + \overline{C}$, then property P.1 holds for the pair $\{A_i, B_i\}$ and, thus, there is no need to add an additional edge to E.

2. If every point q in B_i is incident on some edge $\{q, r\}$ with $r \in q + \overline{C'}$, then property P.2 holds for $\{A_i, B_i\}$ and, again, there is no need to add an additional edge to E.

3. Otherwise, we pick an arbitrary point x in $R^\lambda(A_i)$ that is not incident on any edge $\{x, r\}$ with $r \in x + \overline{C}$ and an arbitrary point y in $R^\lambda(B_i)$ that is not incident on any edge $\{y, r\}$ with $r \in y + \overline{C'}$ and add the edge $\{x, y\}$ to E. The addition of this edge guarantees that property P.3 holds for the pair $\{A_i, B_i\}$. Furthermore, the new edge $\{x, y\}$ makes an angle of at least θ with all edges in the old set E that are incident on x or y.

In the next section, we will present a detailed description of the collection $\mathcal{C}$ of cones and prove the relevant properties that were mentioned above. In Sections 5 and 6, we will use this collection of cones to present our algorithm for computing an angle-constrained spanner.

4 Simplicial Cones

Let V be a set of d linearly independent points in $\mathbb{R}^d$. The set

$$C = \left\{ \sum_{v \in V} \mu_v v : \mu_v \geq 0 \text{ for all } v \in V \right\}$$

is called a *simplicial cone* with *apex* at the origin 0. If we define r_v to be the infinite ray emanating from the origin and going through v, then this cone is equal to the convex hull of the rays r_v, where v ranges over all elements of V. Thus, C is bounded by d hyperplanes, each one containing the origin. The *angular diameter* of C is defined to be

$$\max \{\angle(0x, 0y) : x, y \in C \setminus \{0\}\}.$$

We fix real numbers ϵ and λ such that $0 < \epsilon \leq \pi/2$ and $\lambda \geq 1$. Let S be a set of n points in $\mathbb{R}^d$, and consider a WSPD $\{A_1, B_1\}, \ldots, \{A_m, B_m\}$ of S with separation ratio s, where

$$s \geq \max\left(8\lambda\sqrt{d}, \frac{4\lambda\sqrt{d}}{\sin\epsilon}\right). \tag{6}$$

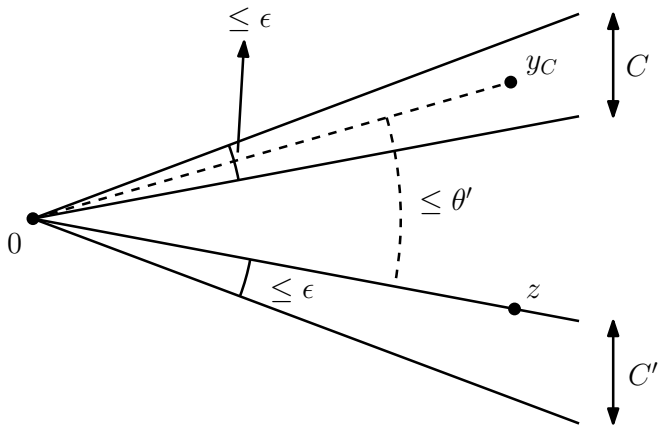

Figure 3: *Since* $\angle(0y_C, 0z) \leq \theta' = \theta + 3\epsilon$, *the cone* C' *is contained in* I_C *and, therefore, in* $\overline{C}$.

For each i with $1 \leq i \leq m$, define ℓ_i to be the distance between the centers of the bounding boxes $R(A_i)$ and $R(B_i)$ of A_i and B_i, respectively.

For each i with $1 \leq i \leq m$, we assume that we are given boxes $R^\lambda(A_i)$ and $R^\lambda(B_i)$, both having diameter at most $2\lambda\sqrt{d}\ell_i/s$, that contain $R(A_i)$ and $R(B_i)$, respectively. Observe that, by Lemma 1, $R^\lambda(A_i)$ and $R^\lambda(B_i)$ exist.

Let $\mathcal{C}$ be a collection of simplicial cones that cover $\mathbb{R}^d$, such that each cone has its apex at the origin and angular diameter at most ϵ. Lukovszki [17] has shown how to obtain such a collection $\mathcal{C}$ consisting of $O(1/\epsilon^{d-1})$ cones in $O(1/\epsilon^{d-1})$ time. Moreover, for any point q in $\mathbb{R}^d$, a cone in $\mathcal{C}$ that contains q can be computed in $O(\log|\mathcal{C}|) = O(\log(1/\epsilon))$ time. (See also Chapter 5 in [19].)

For each cone C in $\mathcal{C}$, we fix an arbitrary point y_C in $C \setminus \{0\}$. Let $\theta > 0$ be a real number and define $\theta' = \theta + 3\epsilon$. For any cone C in $\mathcal{C}$, define

$$I_C = \{C' \in \mathcal{C} : \exists z \in C' \setminus \{0\}, \angle(0y_C, 0z) \leq \theta'\}$$

and

$$\overline{C} = \bigcup_{C' \in I_C} C';$$

see Figure 3. Observe that for any point r in $\overline{C} \setminus \{0\}$, $\angle(0y_C, 0r) \leq \theta' + \epsilon$.

In the rest of this section, we will prove the properties about the cones C, the corresponding sets $\overline{C}$, and the WSPD that were mentioned at the end of Section 3.

Lemma 5. Consider a pair $\{A_i, B_i\}$ in the WSPD, let x be a point in the box $R^\lambda(A_i)$, and let y and y' be points in the box $R^\lambda(B_i)$. Then

1. $|xy| \geq \ell_i/2$ and
2. $\angle(xy, xy') \leq \epsilon$.

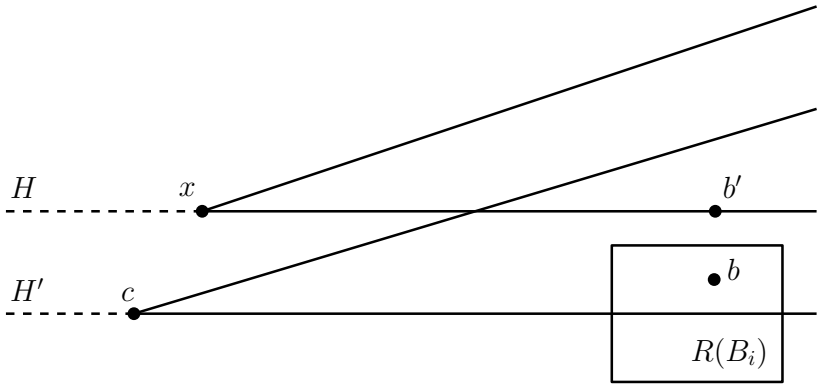

Figure 4: *An illustration of the proof of Lemma 6.*

Proof. Let c and c' be the centers of $R(A_i)$ and $R(B_i)$, respectively. Since both $|cx|$ and $|c'y|$ are at most $2\lambda\sqrt{d}\ell_i/s$, we have

$$\ell_i = |cc'| \leq |cx| + |xy| + |yc'| \leq 4\lambda\sqrt{d}\ell_i/s + |xy| \leq \ell_i/2 + |xy|.$$

It follows that $|xy| \geq \ell_i/2$, proving the first claim.

Since $|yy'| \leq 2\lambda\sqrt{d}\ell_i/s$, it follows that

$$\sin\angle(xy, xy') \leq \frac{|yy'|}{|xy|} \leq \frac{2\lambda\sqrt{d}\ell_i/s}{\ell_i/2} = \frac{4\lambda\sqrt{d}}{s} \leq \sin\epsilon,$$

where the last inequality follows from (6). Since $0 < \epsilon \leq \pi/2$, the second claim follows. □

Lemma 6. Consider a pair $\{A_i, B_i\}$ in the WSPD, let c be the center of the bounding box $R(A_i)$ of A_i, and let C be a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps the translated cone $c + C$. Let b be a point in the intersection of $R(B_i)$ and $c + C$, and let x be a point in $R^\lambda(A_i)$. Then there exists a point b' in the translated cone $x + C$ such that $b' \neq x$ and $\angle(xb, xb') \leq \epsilon$.

Proof. If b is contained in $x + C$, then we take $b' = b$. In this case, the lemma clearly holds.

Assume that b is not contained in $x + C$. We define b' to be the point in $x + C$ for which $|bb'|$ is minimum; see Figure 4 for an illustration.

Let H be the hyperplane through b' having normal $b' - b$ (i.e., the vector from b to b'). Recall that, since C is a simplicial cone, the translated cone $x + C$ is bounded by d hyperplanes, each one containing the apex x. Since b is outside of $x + C$ and b' is the point in $x + C$ closest to b, the hyperplane H is tangent to $x + C$. In particular, (i) b and $x + C$ are on opposite sides of H, (ii) H contains the apex x of the cone $x + C$, and (iii) H contains the line through x and b'.

We may assume without loss of generality that b is "below" H and $x + C$ is "above" H. Let H' be the hyperplane through c that is parallel to H. We observe that $c + C$ is above H'. Therefore, since b is contained in $c + C$, b is also above H'.

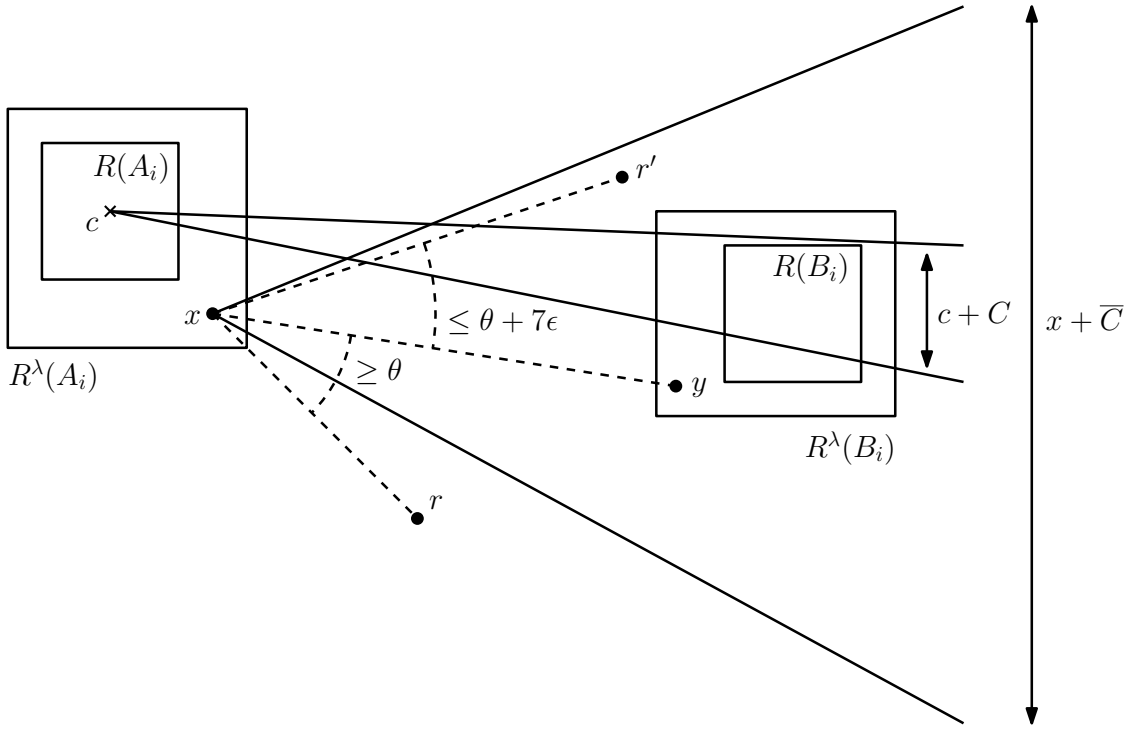

Figure 5: *Illustrating Lemmas 7 and 8.*

Thus, H and H' are two parallel hyperplanes such that (i) $x \in H$, (ii) $c \in H'$, (iii) b is between H and H', and (iv) b' is the point in H that is closest to b. It follows that

$$|bb'| \leq |cx| \leq 2\lambda\sqrt{d}\ell_i/s.$$

In particular, by our choice of s (see (6)), we have $|bb'| \leq \ell_i/4$. Since $|xb| \geq \ell_i/2$ (by Lemma 5), it follows that $b' \neq x$. To complete the proof, we have

$$\sin\angle(xb, xb') = \frac{|bb'|}{|xb|} \leq \frac{2\lambda\sqrt{d}\ell_i/s}{\ell_i/2} = 4\lambda\sqrt{d}/s \leq \sin\epsilon,$$

where the last inequality follows from (6). Since $0 < \epsilon \leq \pi/2$, it follows that $\angle(xb, xb') \leq \epsilon$. □

For the following two lemmas, refer to Figure 5.

Lemma 7. Consider a pair $\{A_i, B_i\}$ in the WSPD, let c be the center of the bounding box $R(A_i)$ of A_i, and let C be a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps the translated cone $c + C$. Let x be a point in $R^\lambda(A_i)$, let y be a point in $R^\lambda(B_i)$, and let r be a point that is not contained in $x + \overline{C}$. Then $\angle(xy, xr) \geq \theta$.

Proof. Consider the point y_C in $C \setminus \{0\}$ and define $y'_C = x + y_C$. Then $\angle(xy'_C, xr) \geq \theta'$, because otherwise, r would be contained in $x + \overline{C}$.

Let b be a point in the intersection of $R(B_i)$ and $c + C$. By Lemma 6, there is a point b' in $x + C$ such that $b' \neq x$ and $\angle(xb', xb) \leq \epsilon$. Since y'_C and b' are in the cone $x + C$, we have $\angle(xy'_C, xb') \leq \epsilon$. Finally, since $x \in R^\lambda(A_i)$ and $b, y \in R^\lambda(B_i)$, we have, by Lemma 5, $\angle(xb, xy) \leq \epsilon$.

By combining these inequalities, we obtain

$$\begin{aligned} \theta' &\leq \angle(xy'_C, xr) \\ &\leq \angle(xy'_C, xb') + \angle(xb', xb) + \angle(xb, xy) + \angle(xy, xr) \\ &\leq 3\epsilon + \angle(xy, xr). \end{aligned}$$

Therefore, $\angle(xy, xr) \geq \theta' - 3\epsilon = \theta$. □

Lemma 8. Consider a pair $\{A_i, B_i\}$ in the WSPD, let c be the center of the bounding box $R(A_i)$ of A_i, and let C be a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps the translated cone $c + C$. Let x be a point in $R^\lambda(A_i)$, let y be a point in $R^\lambda(B_i)$, and let r' be a point that is contained in $x + \overline{C}$. Then $\angle(xy, xr') \leq \theta + 7\epsilon$.

Proof. Consider the point y_C in $C \setminus \{0\}$ and define $y'_C = x + y_C$. Since r' is contained in $x + \overline{C}$, we have $\angle(xy'_C, xr') \leq \theta' + \epsilon$.

Let b be a point in the intersection of $R(B_i)$ and $c + C$. By Lemma 6, there is a point b' in $x + C$ such that $b' \neq x$ and $\angle(xb, xb') \leq \epsilon$. Since b' and y'_C are contained in the cone $x + C$, we have $\angle(xb', xy'_C) \leq \epsilon$. Finally, since $x \in R^\lambda(A_i)$ and $b, y \in R^\lambda(B_i)$, we have, by Lemma 5, $\angle(xy, xb) \leq \epsilon$.

By combining these inequalities, we obtain

$$\begin{aligned} \angle(xy, xr') &\leq \angle(xy, xb) + \angle(xb, xb') + \angle(xb', xy'_C) + \angle(xy'_c, xr') \\ &\leq \theta' + 4\epsilon \\ &= \theta + 7\epsilon. \end{aligned}$$

□

5 A Preliminary Algorithm

In this section, we present a preliminary algorithm that computes an angle-constrained spanner. In Section 6, we will see that a variant of this algorithm can be implemented so that its running time is $O(n \log n)$.

The input to the algorithm is a set S of n points in $\mathbb{R}^d$ and two real numbers θ and ϵ such that

$$0 < \theta < \pi/3 \text{ and } 0 < \epsilon < (\pi - 3\theta)/(21 + \pi).$$

Let $\mathcal{C}$ be a collection of $O(1/\epsilon^{d-1})$ simplicial cones of angular diameter at most ϵ; see Section 4. Recall how we defined $\overline{C}$ for every cone C in $\mathcal{C}$. Let

$$s = \max\left(8\sqrt{d}, \frac{4\sqrt{d}}{\sin\epsilon}, \frac{4}{\sqrt{1+\epsilon} - 1}, 8 + \frac{16}{\epsilon}\right).$$

Step 1: Compute a WSPD $\{A_1, B_1\}, \ldots, \{A_m, B_m\}$ for S with separation ratio s, where $m = O(s^d n)$. For each i with $1 \leq i \leq m$, let ℓ_i be the distance between the centers of the bounding boxes $R(A_i)$ and $R(B_i)$. Sort the pairs in the WSPD according to the values of ℓ_i. Renumber the pairs so that $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$.

Step 2: Initialize an empty edge set E.

Step 3: Process the pairs in the WSPD in increasing order of their indices. Let $\{A_i, B_i\}$ be the current pair to be processed.

1. Let c be the center of $R(A_i)$, let C be a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps the cone $c + C$, and let $L(A_i)$ be the set of all points p in A_i such that the current edge set E does not contain any edge $\{p, r\}$ with $r \in p + \overline{C}$.
2. Let c' be the center of $R(B_i)$, let C' be a cone in $\mathcal{C}$ such that $R(A_i)$ overlaps the cone $c' + C'$, and let $L(B_i)$ be the set of all points q in B_i such that the current edge set E does not contain any edge $\{q, r\}$ with $r \in q + \overline{C'}$.
3. If both $L(A_i)$ and $L(B_i)$ are non-empty, choose an arbitrary point x in $L(A_i)$ and an arbitrary point y in $L(B_i)$, and add the edge $\{x, y\}$ to E.

Step 4: Return the graph $G = (S, E)$.

The next two lemmas state that this algorithm returns an angle-constrained spanner. Their proofs use the results of Sections 3 and 4. In order to show that these results can be applied, we define $\alpha = \theta + 7\epsilon$ and $\lambda = 1$.

First observe that $0 < \alpha < \pi/3$, $\lambda \geq 1$, and $s > 8\lambda$. We will show below that $0 < \epsilon < 2\cos\alpha - 1$. Therefore, the conditions in (1) are satisfied and, thus, the results in Section 3 can indeed be applied.

The line through $(0, 1)$ and $(\pi/3, 1/2)$ has equation $y = 1 - 3x/(2\pi)$. For $0 < x < \pi/3$, the function $y = \cos x$ is above this line. Thus, the inequality $0 < \epsilon < 2\cos\alpha - 1$ is satisfied if we can show that $1 - 3\alpha/(2\pi) > (1 + \epsilon)/2$. By substituting α and rewriting the inequality, we obtain the equivalent inequality $\epsilon < (\pi - 3\theta)/(21 + \pi)$, which is satisfied by our choices for ϵ and θ.

We define, for each i with $1 \leq i \leq m$, $R^\lambda(A_i) = R(A_i)$ and $R^\lambda(B_i) = R(B_i)$. Observe that, by Lemma 1, the diameters of $R(A_i)$ and $R(B_i)$ are at most $2\sqrt{d}\ell_i/s$. Finally, observe that the restriction on the separation ratio s in (6) is satisfied. Thus, the results in Section 4 can be applied.

Lemma 9. The graph $G = (S, E)$ that is returned by the above algorithm is θ-angle-constrained.

Proof. Consider an edge $\{x, y\}$ that is added to the edge set E during the processing of the pair $\{A_i, B_i\}$. During the processing of this pair, the algorithm chooses a cone C in $\mathcal{C}$ such that $R(B_i)$ overlaps $c + C$, where c is the center of $R(A_i)$. It follows from the algorithm that $x \in L(A_i)$ and $y \in L(B_i)$ and, therefore, $x \in A_i$ and $y \in B_i$. Furthermore, just before

$\{x, y\}$ was added to E, there was no edge $\{x, r\}$ in E with $r \in x + \overline{C}$. It follows from Lemma 7 that $\{x, y\}$ makes an angle of at least θ with all edges incident on x that were previously added to the edge set E. By a symmetric argument, $\{x, y\}$ makes an angle of at least θ with all edges incident on y that were previously added to E. □

Lemma 10. The graph $G = (S, E)$ that is returned by the above algorithm is a t-spanner, where

$$t = \frac{1 + \sqrt{2(1+\epsilon)(1-\cos(\theta+7\epsilon)) + \epsilon^2}}{2\cos(\theta+7\epsilon) - 1 - \epsilon}.$$

Proof. Recall that $\alpha = \theta + 7\epsilon$ and $\lambda = 1$. We will prove that the assumption in Lemma 4 holds. Since, by our choice of s, $(s+8)/(s-8) \leq 1+\epsilon$, this, together with Lemma 2, will imply the lemma.

Let i be an integer with $1 \leq i \leq m$ and consider the iteration in which the pair $\{A_i, B_i\}$ is processed. Consider the sets $L(A_i)$ and $L(B_i)$ in Step 3. There are three possible cases.

Case 1: $L(A_i) = \emptyset$.

We will show that property P.1 holds for the pair $\{A_i, B_i\}$. Recall from the algorithm that c is the center of $R(A_i)$ and C is a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps the cone $c + C$.

Let p be an arbitrary point in A_i. Since $L(A_i) = \emptyset$, the edge set E contains an edge $\{p, r\}$ with $r \in p + \overline{C}$. Let q be an arbitrary point in B_i. By Lemma 8, we have $\angle(pq, pr) \leq \theta + 7\epsilon = \alpha$.

It remains to show that $|pr| \leq (1+\epsilon)|pq|$. Let j be the index such that the edge $\{p, r\}$ was added to E during the processing of the pair $\{A_j, B_j\}$. Since this pair was processed before $\{A_i, B_i\}$, we have $\ell_j \leq \ell_i$. It follows from the algorithm that (i) $p \in A_j$ and $r \in B_j$ or (ii) $p \in B_j$ and $r \in A_j$. Combining this with Lemma 1, we obtain

$$|pr| \leq (1 + 4/s)\ell_j \leq (1 + 4/s)\ell_i \leq (1 + 4/s)^2|pq|.$$

By our choice of s, we have $(1 + 4/s)^2 \leq 1 + \epsilon$.

Case 2: $L(B_i) = \emptyset$.

Using an argument that is symmetric to the one for Case 1, we can prove that property P.2 holds for the pair $\{A_i, B_i\}$.

Case 3: Both $L(A_i)$ and $L(B_i)$ are non-empty.

We will show that property P.3 holds for the pair $\{A_i, B_i\}$. Consider the edge $\{x, y\}$ that is added to E during the processing of the pair $\{A_i, B_i\}$. Observe that $x \in A_i$ and $y \in B_i$. Since, by Lemma 1, both A_i and B_i have diameter at most $(2/s)\ell_i$, property P.3 is satisfied. □

Let us consider the running time of this algorithm. First, the collection $\mathcal{C}$ of cones can be computed in $O(1/\epsilon^{d-1})$ time. It takes $O(n \log n + s^d n)$ time to compute the WSPD.

Sorting the $m = O(s^d n)$ pairs takes

$$O(m \log m) = O(m \log n) = O(s^d n \log n)$$

time, where we have used the fact that $m \leq \binom{n}{2}$. Thus, Step 1 takes $O(s^d n \log n)$ time. Step 2 takes $O(1)$ time.

A naive implementation of Step 3 has a running time which is proportional to $\sum_{i=1}^{m}(|A_i|+|B_i|)$. This summation can be as large as $\Theta(n^2)$; see Callahan and Kosaraju [7].

We improve the running time for Step 3 by maintaining, for each cone C in $\mathcal{C}$, a range tree RT_C storing all points p of S for which the current edge set E does not contain any edge $\{p, r\}$ with $r \in p + \overline{C}$. At the start of Step 3, each tree RT_C stores all points of S. Consider the iteration in Step 3 in which the pair $\{A_i, B_i\}$ is processed. Deciding whether $L(A_i)$ and $L(B_i)$ are both non-empty can be done by performing two range emptiness queries: One in RT_C with the bounding box $R(A_i)$ of A_i and the other in $RT_{C'}$ with the bounding box $R(B_i)$ of B_i. (Recall from Section 2 that $A_i = S \cap R(A_i)$ and $B_i = S \cap R(B_i)$.) If the algorithm adds an edge $\{x, y\}$ to the edge set E, then x is deleted from all range trees $RT_{C''}$ for which $y \in x + \overline{C''}$, and y is deleted from all range trees $RT_{C''}$ for which $x \in y + \overline{C''}$.

Thus, it is sufficient to use range trees that support range emptiness queries and deletions. Mehlhorn and Näher [18] have shown that both of these operations can be done in $O(\log^{d-1} n)$ time. Moreover, such a range tree can be built in $O(n \log^{d-1} n)$ time. It follows that the total running time for Step 3 is the sum of

- $O((1/\epsilon^{d-1}) n \log^{d-1} n)$, which is the total time to build $O(1/\epsilon^{d-1})$ range trees (one for each cone),
- $O(s^d n \log^{d-1} n)$, which is the total time for $2m = O(s^d n)$ range emptiness queries, and
- $O((1/\epsilon^{d-1}) n \log^{d-1} n)$, which is the total time to perform n deletions in each range tree.

Thus, the overall running time of the algorithm is

$$O\left(\left(s^d + (1/\epsilon^{d-1})\right) n \log^{d-1} n\right).$$

Note that for small values of ϵ, the separation ratio s is proportional to $1/\epsilon$, leading to a running time of $O((1/\epsilon^d) n \log^{d-1} n)$.

This result has, however, two drawbacks. First, the amount of space used by the algorithm is $O((1/\epsilon^{d-1}) n \log^{d-1} n)$. Second, the algorithms in [18] do not work in the algebraic computation-tree model. Thus, even though the running time is $O(n \log n)$ in the case when $d = 2$, the $\Omega(n \log n)$ lower bound of [8] on the time to compute any spanner does not apply.

In the next section, we show that a variant of the algorithm can be implemented to run in optimal $O(n \log n)$ time.

6 An Optimal Algorithm

6.1 The Main Idea

Consider again the WSPD $\{A_1, B_1\}, \ldots, \{A_m, B_m\}$ of the point set S. Das *et al.* [10] (see also Arya *et al.* [2]) define, for each pair $\{A_i, B_i\}$, the *dumbbell* D_i to be the geometric object consisting of the bounding boxes $R(A_i)$ and $R(B_i)$, together with the line segment joining the centers of these boxes. The two boxes $R(A_i)$ and $R(B_i)$ are called the *heads* of the dumbbell. The *length* of the dumbbell D_i is defined to be the distance between the centers of its heads. Thus, using our previous notation, the length of D_i is equal to ℓ_i.

In the algorithm of Section 5, it is crucial that the pairs $\{A_i, B_i\}$ (or, equivalently, the dumbbells D_i) are processed in non-decreasing order of their lengths; see Case 1 in the proof of Lemma 10. Assume that for any two dumbbells, either their four heads are pairwise disjoint, or one dumbbell is completely contained in the head of the other dumbbell. Then we can store the dumbbells at the nodes of a "nesting tree" such that, for every dumbbell D_i, all dumbbells that are completely contained in either of its heads are stored in the subtree of the node storing D_i. In particular, each dumbbell in the subtree of D_i has a length that is less than ℓ_i.

During the processing of the dumbbells, each node v for which (i) its dumbbell has not been processed but (ii) all dumbbells in its subtree have been processed, stores $O(1)$ lists: Let D_i be the dumbbell stored at v. Then, for every cone C in $\mathcal{C}$, the node v stores two lists $L_A(v, C)$ and $L_B(v, C)$. The list $L_A(v, C)$ stores all points $p \in A_i$ such that (i) p is in some dumbbell stored in the subtree of v and (ii) the current graph does not have any edge $\{p, r\}$ with $r \in p + \overline{C}$. The list $L_B(v, C)$ stores a subset of B_i and is similarly defined.

These lists allow us to implement Step 3 of the algorithm in $O(n)$ time, because processing the dumbbells in sorted order implies a bottom-up traversal of the nesting tree.

Unfortunately, the dumbbells do not have this nesting property. Arya *et al.* [2], however, have shown that the dumbbells can be partitioned into $O(1)$ groups, such that the nesting property "almost" holds for each group: Each group can be stored in a "dumbbell tree" such that the following holds: For any dumbbell D_i, all dumbbells in its subtree are much shorter than D_i and very close to D_i. Thus, even though there may be a point p in the subtree of D_i that is not contained in either of its heads, p is still close to one of the heads. As we will prove later, this implies that, by using a value of λ that is larger than 1, we can still apply the results in Section 3.

In the next section, we briefly recall those ingredients of the dumbbell trees that are relevant for our algorithm.

6.2 Dumbbell Trees

As mentioned above, the dumbbell trees are due to Arya *et al.* [2]. Since they did not provide the full details of this construction, in particular, the dependence on the different parameters, we will follow the exposition in Chapter 11 of [19]; in particular, refer to Section 11.9.

Let $s > 1$ be the separation ratio and consider the WSPD $\{A_1, B_1\}, \ldots, \{A_m, B_m\}$ of the point set S, where $m = O(s^d n)$, and the corresponding set $\mathcal{D}$ of dumbbells $D_1, \ldots, D_m$. Let β and γ be real numbers such that

$$0 < \beta < \min\left(\frac{1}{2}, \frac{s}{\sqrt{d}(s+4)}\right), \tag{7}$$

$$6\beta + \frac{8\sqrt{d}}{s} < \min\left(\gamma, \frac{s}{s+4}\right), \tag{8}$$

and

$$\beta\left(1 + 2\gamma + \frac{2\sqrt{d}}{s}\right) \leq \gamma < 1. \tag{9}$$

Let R_0 be a large box that contains all dumbbells of $\mathcal{D}$. We define a *dummy dumbbell* D_0 whose heads are R_0 and a translated copy R_0' of R_0 such that the distance between the centers of R_0 and R_0' (and, thus, the length ℓ_0 of D_0) is equal to $1/\beta$ times the maximum length of any dumbbell in $\mathcal{D}$. In

$$O\left(s^d(1+\gamma s)^d(\log(1/\beta))n\log n + s^{2d}(1+\gamma s)^d n\right) \tag{10}$$

total time, the set $\mathcal{D}$ of dumbbells can be partitioned into

$$k = O\left(s^d(1+\gamma s)^d \log(1/\beta)\right)$$

groups $\mathcal{D}_1, \ldots, \mathcal{D}_k$, and for each group, a *dumbbell tree* can be constructed. For any index ℓ with $1 \leq \ell \leq k$, each node in the dumbbell tree T_ℓ is either

1. a *dumbbell node*, in which case it stores a dumbbell of the set $\mathcal{D}_\ell \cup \{D_0\}$,
2. a *head node*, in which case it stores a head of some dumbbell in the set $\mathcal{D}_\ell \cup \{D_0\}$, or
3. a *leaf*, in which case it stores a point of S.

The tree T_ℓ has the following properties (see Figure 6):

1. Each dumbbell in $\mathcal{D}_\ell \cup \{D_0\}$ is stored at a unique dumbbell node, each head of each dumbbell in this set is stored at a unique head node, and each point of S is stored at a unique leaf.
2. The root is a dumbbell node and stores the dummy dumbbell D_0.
3. Each dumbbell node storing a dumbbell D_i, has two children, which are head nodes storing the heads of D_i.
4. Each head node v storing a head R of a dumbbell D_i, has dumbbell nodes and leaves as children:

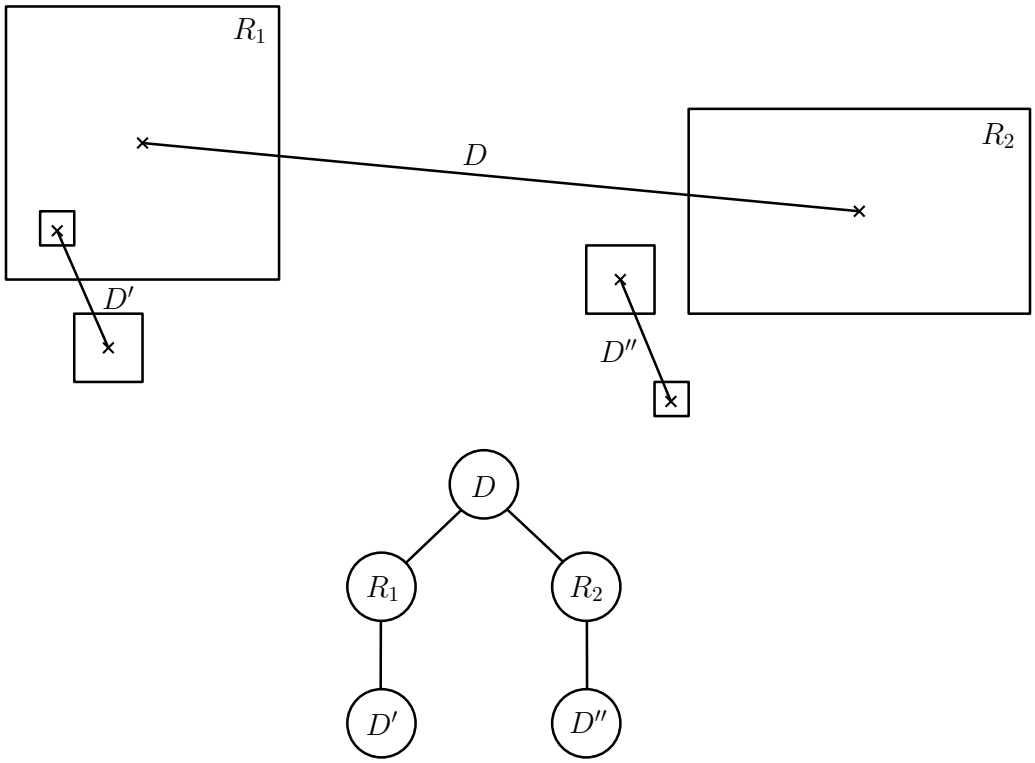

Figure 6: *The dumbbell D is stored at a dumbbell node, whose children are head nodes, storing the heads R_1 and R_2 of D. The dumbbell D' is "much shorter" than D and "close" to R_1; therefore, the dumbbell node storing D' is a child of the head node storing R_1. Similarly, the dumbbell node storing D'' is a child of the head node storing R_2.*

(a) If a child is a dumbbell node storing a dumbbell D_j, then
   i. $\ell_j \leq \beta \ell_i$ and
   ii. the distance of closest approach between D_j and the head R is at most $\gamma \ell_j$.

(b) If a child is a leaf storing a point p, then $p \in R$ and p is not contained in any of the heads that are stored in the proper subtree of v.

The following property, which is Lemma 11.8.2 in [19], will be important for us.

Lemma 11. Consider a dumbbell tree T_ℓ. Let p be a point of S, let u be the leaf in T_ℓ that stores p, and let v be a head node in T_ℓ whose head contains p. Then u is in the subtree of v.

In the rest of this section, we will take

$$s \geq 32\sqrt{d}, \ \beta = 1/(4s), \text{ and } \gamma = 1/2. \tag{11}$$

It is not difficult to verify that the conditions in (7), (8), and (9) are satisfied.

The following lemma states that for every head node v, all points in the subtree of v are "close" to the head stored at v. This will allow us to use the results of Section 3 with the value $\lambda = 2$.

Lemma 12. Let v be a head node of a dumbbell tree, let R be the head stored at v, and let D_i be the dumbbell stored at the parent of v (thus, R is a head of D_i). Let S_v be the set of all points in S that are stored at the leaves of the subtree of v. Then

1. the diameter of S_v is at most $4\ell_i/s$,
2. the bounding box of S_v has diameter at most $4\sqrt{d}\ell_i/s$.

Proof. First observe that each point of S_v is contained either in R or in one of the heads of one of the dumbbells that is stored in the proper subtree of v.

Consider two dumbbells D and D' in the subtree of v such that the node storing D is the grandparent of the node storing D'. Let ℓ and ℓ' be the lengths of D and D', respectively. By the properties of the dumbbell trees, $\ell' \leq \beta\ell$ and the distance of closest approach between D and D' is at most $\gamma\ell'$. By Lemma 1, the diameter of D' is at most $(1 + 4/s)\ell'$. Therefore, any point of D' is within distance

$$\gamma\ell' + (1 + 4/s)\ell' \leq \beta(\gamma + 1 + 4/s)\ell \leq \ell$$

of some point of D, where the last inequality follows from our choices of s, β, and γ in (11).

The argument above implies that any point in S_v is within distance

$$\sum_{j=1}^{\infty} \beta^j \ell_i = \frac{\beta}{1-\beta}\,\ell_i \leq \ell_i/s$$

of some point of the head R. Since, by Lemma 1, the diameter of R is at most $2\ell_i/s$, it follows that the diameter of S_v is at most $4\ell_i/s$. This proves the first claim. The second claim follows from the fact that the length of each side of the bounding box of S_v is at most $4\ell_i/s$. □

6.3 The Algorithm

We are now ready to present the final algorithm. The input is a set S of n points in $\mathbb{R}^d$ and real numbers θ and ϵ such that

$$0 < \theta < \pi/3 \text{ and } 0 < \epsilon < (\pi - 3\theta)/(21 + \pi).$$

Consider again a collection $\mathcal{C}$ of $O(1/\epsilon^{d-1})$ simplicial cones of angular diameter at most ϵ; see Section 4. Recall how we defined $\overline{C}$ for every cone C in $\mathcal{C}$. Let

$$s = \max\left(32\sqrt{d}, \frac{8\sqrt{d}}{\sin\epsilon}, \frac{8}{\sqrt{1+\epsilon}-1}, 16 + \frac{32}{\epsilon}\right). \tag{12}$$

The algorithm starts by computing a WSPD $\{A_i, B_i\}$, $1 \leq i \leq m = O(s^d n)$, for S with separation ratio s, the corresponding dumbbells $D_1, \ldots, D_m$, and the dumbbell trees $T_1, \ldots, T_k$, where $k = O(s^{2d} \log s)$. (Note that, since $s \geq 32\sqrt{d}$, we can apply the results of Section 6.2; see (11).)

We assume that the pairs in the WSPD have been sorted so that $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$. As in Section 5, the algorithm processes the pairs (or, equivalently, the dumbbells) in this order. Observe that if a dumbbell has been processed, all dumbbells in its subtree have also been processed. The algorithm will maintain the following invariant:

Invariant: Every head node v of every dumbbell tree stores lists $L(v, C)$, where C ranges over all cones in $\mathcal{C}$. Let S_v be the set of all points in S that are stored at the leaves of v's subtree. If (i) the dumbbell stored at the parent of v has not been processed but (ii) all dumbbells in the subtree of v have been processed, then, for each C in $\mathcal{C}$, the list $L(v, C)$ stores all points p in S_v for which the current edge set E does not contain any edge $\{p, r\}$ with $r \in p + \overline{C}$.

Let p be a point in S and let C be a cone in $\mathcal{C}$. Each dumbbell tree has at most one head node v satisfying (i) and (ii), and such that $p \in S_v$. Therefore, there are at most k lists $L(\cdot, C)$ that contain p. In order to avoid searching for p in such a list, we do the following: For each point p in S and for each cone C in $\mathcal{C}$, we store a list $LP(p, C)$ of pointers to the positions of p in all lists $L(\cdot, C)$ in which p occurs.

Initialization:

1. For every head node v of every dumbbell tree, and for every cone C in $\mathcal{C}$, initialize an empty list $L(v, C)$.
2. For every point p in S and for every cone C in $\mathcal{C}$, initialize an empty list $LP(p, C)$.
3. For every leaf w of every dumbbell tree, and for every cone C in $\mathcal{C}$, do the following: Let p be the point stored at w, and let v be the parent of w. Add the point p to the list $L(v, C)$, and add, to $LP(p, C)$, a pointer to the position of p in $L(v, c)$.
4. Initialize an empty edge set E.

It is easy to verify that the invariant holds after the initialization. Now the algorithm processes the pairs $\{A_i, B_i\}$ in the WSPD in increasing order of their indices.

Processing the pair $\{A_i, B_i\}$: Let u be the dumbbell node storing D_i, and let v and w be the children of u storing the heads $R(A_i)$ and $R(B_i)$ of D_i, respectively.

1. Let c be the center of $R(A_i)$ and let C be a cone in $\mathcal{C}$ such that $R(B_i)$ overlaps the cone $c + C$.
2. Let c' be the center of $R(B_i)$ and let C' be a cone in $\mathcal{C}$ such that $R(A_i)$ overlaps the cone $c' + C'$.
3. If both $L(v, C)$ and $L(w, C')$ are non-empty, do the following:

(a) Choose an arbitrary point x in $L(v, C)$ and an arbitrary point y in $L(w, C')$, and add the edge $\{x, y\}$ to E.

(b) For each cone C'' in $\mathcal{C}$ for which $y \in x + \overline{C''}$, use the list $LP(x, C'')$ to locate and delete x from all lists $L(\cdot, C'')$ in which x occurs. Set $LP(x, C'')$ to the empty list.

(c) For each cone C'' in $\mathcal{C}$ for which $x \in y + \overline{C''}$, use the list $LP(y, C'')$ to locate and delete y from all lists $L(\cdot, C'')$ in which y occurs. Set $LP(y, C'')$ to the empty list.

4. Let u' be the parent of u. For each cone C'' in $\mathcal{C}$, set $L(u', C'')$ as the concatenation of the lists $L(u', C'')$, $L(v, C'')$, and $L(w, C'')$, and set $L(v, C'')$ and $L(w, C'')$ as empty lists.

It is not difficult to verify that the invariant is maintained during the processing of the pair $\{A_i, B_i\}$. After all pairs have been processed, the algorithm returns the graph $G = (S, E)$.

In the next two lemmas, we will prove that this graph G is an angle-constrained spanner. The proofs are similar to those of Lemmas 9 and 10, and they use the results of Sections 3 and 4. Define $\alpha = \theta + 7\epsilon$ and $\lambda = 2$. We first observe that the conditions in (1) are satisfied and, thus, the results in Section 3 can indeed be applied.

Let i be an index with $1 \leq i \leq m$, and consider the pair $\{A_i, B_i\}$ in the WSPD. Let v be the head node that stores the head $R(A_i)$ of the dumbbell D_i. We define $R^\lambda(A_i)$ to be the bounding box of the set S_v of points stored in the subtree of v. By Lemmas 11 and 12, $R^\lambda(A_i)$ contains $R(A_i)$ and has diameter at most $2\lambda\sqrt{d}\ell_i/s$. The box $R^\lambda(B_i)$ is defined in the same way with respect to B_i. We finally observe that the restriction on the separation ratio s in (6) is satisfied. Thus, the results of Section 4 can be applied.

Lemma 13. The graph $G = (S, E)$ that is returned by the above algorithm is θ-angle-constrained.

Proof. Consider an edge $\{x, y\}$ that is added to the edge set E during the processing of the pair $\{A_i, B_i\}$. Consider the head nodes v and w that store the heads $R(A_i)$ and $R(B_i)$ of the dumbbell D_i, respectively. The algorithm chooses cones C and C' in $\mathcal{C}$ such that $R(B_i)$ overlaps $c + C$, and $R(A_i)$ overlaps $c' + C'$, where c and c' are the centers of $R(A_i)$ and $R(B_i)$, respectively.

It follows from the algorithm that $x \in L(v, C)$ and $y \in L(w, C')$ and, therefore, $x \in S_v$ and $y \in S_w$. Thus, $x \in R^\lambda(A_i)$ and $y \in R^\lambda(B_i)$. Since $x \in L(v, C)$, just before $\{x, y\}$ was added to E, there was no edge $\{x, r\}$ in E with $r \in x + \overline{C}$. It then follows from Lemma 7 that $\{x, y\}$ makes an angle of at least θ with all edges incident on x that were previously added to the edge set E. By a symmetric argument, $\{x, y\}$ makes an angle of at least θ with all edges incident on y that were previously added to E. □

Lemma 14. The graph $G = (S, E)$ that is returned by the above algorithm is a t-spanner, where

$$t = \frac{1 + \sqrt{2(1+\epsilon)(1-\cos(\theta+7\epsilon)) + \epsilon^2}}{2\cos(\theta+7\epsilon) - 1 - \epsilon}.$$

Proof. Recall that $\alpha = \theta + 7\epsilon$ and $\lambda = 2$. We will prove that the assumption in Lemma 4 holds. Since, by our choice of s, $(s+16)/(s-16) \leq 1+\epsilon$, this, together with Lemma 2, will imply the lemma.

Consider the iteration in which the pair $\{A_i, B_i\}$ is processed, and consider the head nodes v and w that store the heads $R(A_i)$ and $R(B_i)$ of the dumbbell D_i, respectively. The algorithm chooses cones C and C' in $\mathcal{C}$ such that $R(B_i)$ overlaps $c + C$ and $R(A_i)$ overlaps $c' + C'$, where c and c' are the centers of $R(A_i)$ and $R(B_i)$, respectively. Consider the lists $L(v, C)$ and $L(w, C')$. There are three possible cases.

Case 1: The list $L(v, C)$ is empty.

We will show that property P.1 holds for the pair $\{A_i, B_i\}$. Let p be an arbitrary point in A_i. Lemma 11 and the invariant imply that $p \in S_v$ and E contains an edge $\{p, r\}$ with $r \in p + \overline{C}$. Let q be an arbitrary point in B_i. Since $p \in R^\lambda(A_i)$ and $q \in R^\lambda(B_i)$, it follows from Lemma 8 that $\angle(pq, pr) \leq \theta + 7\epsilon = \alpha$.

It remains to show that $|pr| \leq (1+\epsilon)|pq|$. Let j be the index such that the edge $\{p, r\}$ was added to E during the processing of the pair $\{A_j, B_j\}$. Since this pair was processed before $\{A_i, B_i\}$, we have $\ell_j \leq \ell_i$. It follows from the algorithm that $p \in R^\lambda(A_j)$ and $r \in R^\lambda(B_j)$ (or vice versa). Thus, using the first claim in Lemma 12, the triangle inequality, and Lemma 1, we obtain

$$\begin{aligned} |pr| &\leq 4\ell_j/s + \ell_j + 4\ell_j/s \\ &= (1 + 8/s)\ell_j \\ &\leq (1 + 8/s)\ell_i \\ &\leq (1 + 8/s)(1 + 4/s)|pq| \\ &\leq (1 + 8/s)^2|pq| \\ &\leq (1+\epsilon)|pq|, \end{aligned}$$

where the last inequality follows from our choice of s.

Case 2: The list $L(w, C')$ is empty.

Using an argument that is symmetric to the one for Case 1, we can prove that property P.2 holds for the pair $\{A_i, B_i\}$.

Case 3: Both lists $L(v, C)$ and $L(w, C')$ are non-empty.

Consider the edge $\{x, y\}$ that is added to E during the processing of the pair $\{A_i, B_i\}$. Then $x \in S_v$ and $y \in S_w$. By Lemmas 11 and 12, $A_i \subseteq S_v$, $B_i \subseteq S_w$, and both S_v and S_w have diameter at most $(2\lambda/s)\ell_i$. Therefore, property P.3 is satisfied. □

We analyze the running time of the algorithm. Recall from (11) that $\beta = 1/(4s)$ and $\gamma = 1/2$, and from (12) that, for small values of ϵ, s is proportional to $1/\epsilon$.

It follows from (10) that the total time to compute the collection $\mathcal{C}$ of $O(1/\epsilon^{d-1})$ cones, the WSPD, and the

$$k = O\left(s^{2d} \log s\right) = O\left(\left(1/\epsilon^{2d}\right) \log(1/\epsilon)\right)$$

dumbbell trees is

$$O\left(\left(1/\epsilon^{2d}\right)(\log(1/\epsilon))n\log n + \left(1/\epsilon^{3d}\right)n\right).$$

The time for the initialization is

$$O\left(kn|\mathcal{C}|\right) = O\left(\left(1/\epsilon^{3d-1}\right)(\log(1/\epsilon))n\right).$$

Consider the processing of a pair $\{A_i, B_i\}$. Steps 1 and 2, i.e., determining the cones C and C', takes $O(\log(1/\epsilon))$ time. The time for Step 3 is proportional to $|\mathcal{C}| = O(1/\epsilon^{d-1})$ plus the number of lists from which x and y are deleted. Obviously, each point of S can be deleted from such a list only once. Therefore, the total time to process all $m = O(s^d n)$ pairs of the WSPD is

$$O\left(m/\epsilon^{d-1}\right) = O\left(\left(1/\epsilon^{2d-1}\right)n\right).$$

We conclude that the total running time of the algorithm is

$$O\left(\left(1/\epsilon^{2d}\right)(\log(1/\epsilon))n\log n + \left(1/\epsilon^{3d}\right)n\right).$$

Thus, we have proved Theorem 1.

7 Concluding Remarks

We have presented an $O(n\log n)$–time algorithm that constructs, for any set of n points in $\mathbb{R}^d$, a t-spanner for S, in which any two edges $\{p,q\}$ and $\{p,r\}$ make an angle of at least θ. The upper bound t on the stretch factor is a function of θ; for values of θ approaching 0, the stretch factor approaches 1. As a corollary, we have obtained an $O(n\log n)$–time algorithm that constructs, for any $t > 1$, a t-spanner of maximum degree $O(1/(t-1)^{d-1})$. Both the running time and the upper bound on the maximum degree are optimal.

By running the path-greedy algorithm of Gudmundsson *et al.* [14] on the spanner of Theorem 1, we obtain, again in $O(n\log n)$ time, an angle-constrained spanner whose weight is proportional to a minimum spanning tree of the point set.

Our results are valid for any θ with $0 < \theta < \pi/3$. We leave it as an open problem to decide if the same results can be obtained for values of θ that are larger than or equal to $\pi/3$.

Acknowledgments

The authors thank Shay Solomon for communicating the lower bound in [12] on the maximum degree of spanners. The authors also thank the referees for their useful comments.

References

[1] I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.

[2] S. Arya, G. Das, D. M. Mount, J. S. Salowe, and M. Smid. Euclidean spanners: short, thin, and lanky. In *Proceedings of the 27th ACM Symposium on the Theory of Computing*, pages 489–498, 1995.

[3] S. Arya and M. Smid. Efficient construction of a bounded-degree spanner with low weight. *Algorithmica*, 17:33–54, 1997.

[4] N. Bonichon, C. Gavoille, N. Hanusse, and L. Perkovic. Plane spanners of maximum degree six. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming*, volume 6198 of *Lecture Notes in Computer Science*, pages 19–30, Berlin, 2010. Springer-Verlag.

[5] P. Bose, P. Carmi, M. Farshi, A. Maheshwari, and M. Smid. Computing the greedy spanner in near-quadratic time. *Algorithmica*, 58:711–729, 2010.

[6] P. B. Callahan and S. R. Kosaraju. Faster algorithms for some geometric graph problems in higher dimensions. In *Proceedings of the 4th ACM-SIAM Symposium on Discrete Algorithms*, pages 291–300, 1993.

[7] P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM*, 42:67–90, 1995.

[8] D. Z. Chen, G. Das, and M. Smid. Lower bounds for computing geometric spanners and approximate shortest paths. *Discrete Applied Mathematics*, 110:151–167, 2001.

[9] K. L. Clarkson. Approximation algorithms for shortest path motion planning. In *Proceedings of the 19th ACM Symposium on the Theory of Computing*, pages 56–65, 1987.

[10] G. Das, P. Heffernan, and G. Narasimhan. Optimally sparse spanners in 3-dimensional Euclidean space. In *Proceedings of the 9th ACM Symposium on Computational Geometry*, pages 53–62, 1993.

[11] G. Das and P. J. Heffernan. Constructing degree-3 spanners with other sparseness properties. *International Journal of Foundations of Computer Science*, 7:121–135, 1996.

[12] M. Elkin and S. Solomon. Steiner shallow-light trees are exponentially lighter than spanning ones. In *Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science*, pages 373–382, 2011.

[13] D. Eppstein. Spanning trees and spanners. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 425–461. Elsevier Science, Amsterdam, 2000.

[14] J. Gudmundsson, C. Levcopoulos, and G. Narasimhan. Fast greedy algorithms for constructing sparse geometric spanners. *SIAM Journal on Computing*, 31:1479–1500, 2002.

[15] J. M. Keil and C. A. Gutwin. Classes of graphs which approximate the complete Euclidean graph. *Discrete & Computational Geometry*, 7:13–28, 1992.

[16] X.-Y. Li. *Wireless Ad Hoc and Sensor Networks*. Cambridge University Press, Cambridge, UK, 2008.

[17] T. Lukovszki. *New Results on Geometric Spanners and Their Applications*. Ph.D. thesis, Department of Computer Science, University of Paderborn, Paderborn, Germany, 1999.

[18] K. Mehlhorn and S. Näher. Dynamic fractional cascading. *Algorithmica*, 5:215–241, 1990.

[19] G. Narasimhan and M. Smid. *Geometric Spanner Networks*. Cambridge University Press, Cambridge, UK, 2007.

[20] J. S. Salowe. Constructing multidimensional spanner graphs. *International Journal of Computational Geometry & Applications*, 1:99–107, 1991.

[21] J. S. Salowe. Euclidean spanner graphs with degree four. *Discrete Applied Mathematics*, 54:55–66, 1994.

[22] J. Soares. Approximating Euclidean distances by small degree graphs. *Discrete & Computational Geometry*, 11:213–233, 1994.

[23] P. M. Vaidya. A sparse graph almost as good as the complete graph on points in K dimensions. *Discrete & Computational Geometry*, 6:369–381, 1991.